Group project [2-3 weeks]

As Twitter using scala, we can start with some dummy twitter-clone. **It should be just API!** If you want to do some frontend, do it as separate project that calls your API.

Split to teams of 2-3 members and notify TA about your decision. TA's telegram: - @cubazis - @VecnaSecrets - @basov_ae

Entities

You can store everything in memory, since it could be a trouble to use persistent storage (but if you want and can use it)

- User
  - User has: id, email, nickname, password
  - id and email should be unique
- Twit
  - Twit should have: id, text, author, submission time
  - id should be unique
  - [Optional] Twit has likes, dislikes, see `Functionality` section too

Functionality

- A guest can register and then sign in (he is User from that moment), sign out
  - For sessions use JWT-tokens, since it is common to use them now, cookies is option too, but JWT is better for reasons
- User can create twits and only edit and destroy his twits
  - [Optional] Implement mentioning of other people by his nickname
- User can subscribe to other users
- User can get his feed that consists of other users twits
- User can get other users feed (to decide if he want to subscribe to that user)
- User can re-tweet other users twits, so subscribers will get this twit (author should be mentioned) in their feeds
  - He also should be able to remove his re-twit
  - If author will remove his twit, all re-twits of that twit should not be visible too
- [Optional] User can like or dislike a twit
  - If he likes a twit and then dislike it, like is removed and dislike is added. This should works vice versa too
  - User can also just remove his like or dislike

Criteria of done

- Code on github
- Project could be built using instructions in your `README.md` in repository
- Make a small report that points out what was done (which functionality is working).
  - If you haven't managed something to work please write why, so we can cover this material

    during the course.
  - Use Postman to test your API and publish your collection, add link in your report
  - [Optional] If you used some additional tools/materials/guides, please mention them in the report, it could be helpful for other students and try to connect the problem you was solving and material that you was using.## Group project [2-3 weeks] As Twitter using scala, we can start with some dummy twitter-clone. **It should be just API!** If you want to do some frontend, do it as separate project that calls your API.

Split to teams of 2-3 members and notify TA about your decision. TA's telegram: - @cubazis - @VecnaSecrets - @basov_ae

Entities

You can store everything in memory, since it could be a trouble to use persistent storage (but if you want and can use it)

- User
  - User has: id, email, nickname, password
  - id and email should be unique
- Twit
  - Twit should have: id, text, author, submission time
  - id should be unique
  - [Optional] Twit has likes, dislikes, see `Functionality` section too

Functionality

- A guest can register and then sign in (he is User from that moment), sign out
  - For sessions use JWT-tokens, since it is common to use them now, cookies is option too, but JWT is better for reasons
- User can create twits and only edit and destroy his twits
  - [Optional] Implement mentioning of other people by his nickname
- User can subscribe to other users
- User can get his feed that consists of other users twits
- User can get other users feed (to decide if he want to subscribe to that user)
- User can re-tweet other users twits, so subscribers will get this twit (author should be mentioned) in their feeds
  - He also should be able to remove his re-twit
  - If author will remove his twit, all re-twits of that twit should not be visible too
- [Optional] User can like or dislike a twit
  - If he likes a twit and then dislike it, like is removed and dislike is added. This should works vice versa too
  - User can also just remove his like or dislike

Criteria of done

- Code on github

- Project could be built using instructions in your `README.md` in repository
- Make a small report that points out what was done (which functionality is working).
  - If you haven't managed something to work please write why, so we can cover this material during the course.
  - Use Postman to test your API and publish your collection, add link in your report
  - [Optional] If you used some additional tools/materials/guides, please mention them in the report, it could be helpful for other students and try to connect the problem you was solving and material that you was using.